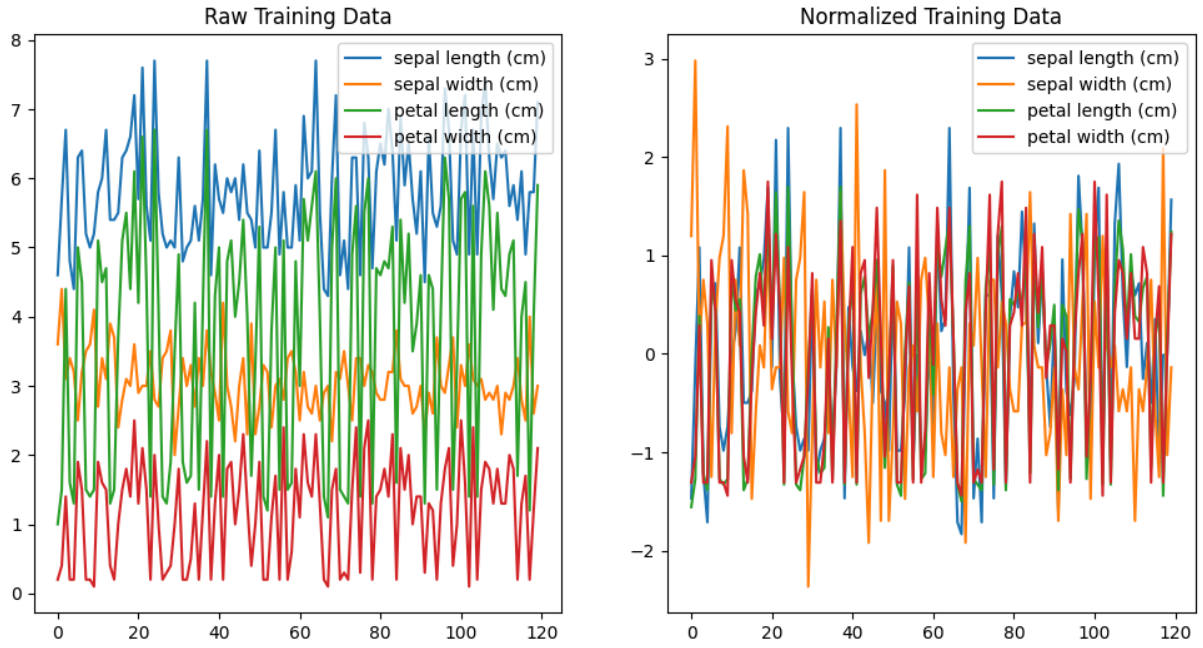
Zeynep Kurtulus
240037253
Assignment1_Part2_Report

Q1. We again notice that the attributes are on different scales. Use the normalization method from last lab, to standardize the scales of each attribute on both sets. Plot the normalized and raw training sets; what do you observe? [2 marks]



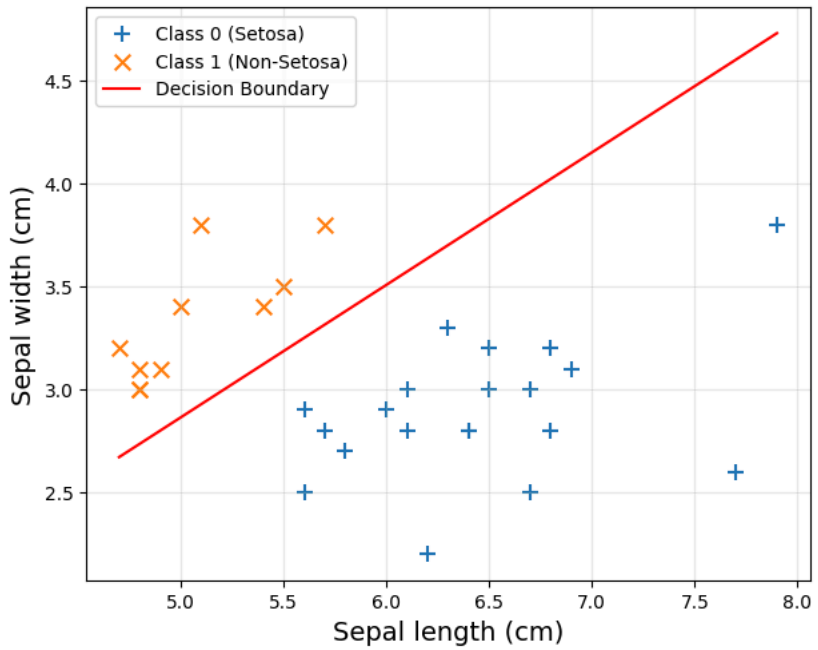
If we look at the figure on the left which represents the raw data, we can see that the attributes of the data are on different scales (each of the attributes have different ranges as can be seen from the figure left). Features with different scales can negatively affect the performance of the gradient descent algorithm. That's where the normalization process comes into the picture, it basically ensures that all features have zero mean and unit variance, making the optimization more efficient and preventing large feature values from dominating the gradient updates.

After applying the normalization method, we end up having the plot on the right. The main difference between the raw and the normalized data is that in the normalized data all the features are not scaled to have similar scales. This standardization ensures that the features are on the same scale, which should make the optimization more efficient because now the algorithm can update all features uniformly rather than having a dominating feature.

So overall, key observations of these two plots are: The raw training data has features with varying ranges, which affects the efficiency of the algorithm. The normalized training data ensures all features are on the same scale which helps prevent any one feature from badly affecting and dominating the model's learning

process. By applying normalization we end up improving the performance of the machine learning model.

Q5. Draw the decision boundary on the test set using the learned parameters. Is this decision boundary separating the classes? Does this match our expectations? [2 marks]



From the plot it can be seen that the decision boundary (red line in the graph) is separating the classes. The Setosa class (blue + symbols) lies mostly below the decision boundary. The Non-Setosa class (orange x symbols) lies above the decision boundary. Yes, this matches our expectations which means that Setosa and Non-Setosa are linearly separable based on the features sepal length and sepal width.
Logistic regression is a linear classifier, so we expect it to have a straight-line decision boundary like this one in the graph. The decision boundary that the model came up with correctly classifies different points.

Q7. Using the 3 classifiers, predict the classes of the samples in the test set and show the predictions in a table. Do you observe anything interesting? [4 marks]

Setosa Confidence	Versicolor Confidence	Virginica Confidence	Predicted Class	Ground Truth Class
0.034410	0.742355	0.457444	Versicolor	Versicolor
0.999964	0.170519	0.110957	Setosa	Setosa
0.000003	0.838924	0.946857	Virginica	Virginica
0.036972	0.590232	0.705695	Virginica	Versicolor
0.009189	0.765599	0.535546	Versicolor	Versicolor
0.999706	0.314759	0.081546	Setosa	Setosa
0.281754	0.547601	0.528553	Versicolor	Versicolor
0.002073	0.376506	0.968881	Virginica	Virginica
0.000522	0.914940	0.389690	Versicolor	Versicolor
0.080001	0.732908	0.374096	Versicolor	Versicolor
0.012816	0.346486	0.947850	Virginica	Virginica
0.999348	0.586137	0.022226	Setosa	Setosa
0.999929	0.302623	0.048840	Setosa	Setosa
0.999542	0.535644	0.026179	Setosa	Setosa
0.999984	0.129791	0.132496	Setosa	Setosa
0.146494	0.341402	0.859967	Virginica	Versicolor
0.000743	0.464648	0.968256	Virginica	Virginica
0.041023	0.828533	0.254797	Versicolor	Versicolor
0.047731	0.672578	0.559298	Versicolor	Versicolor
0.000338	0.577981	0.954167	Virginica	Virginica
0.999661	0.423460	0.047015	Setosa	Setosa
0.013709	0.475620	0.886715	Virginica	Virginica
0.999755	0.281368	0.100008	Setosa	Setosa
0.000457	0.604317	0.939110	Virginica	Virginica
0.018349	0.207955	0.978203	Virginica	Virginica
0.001167	0.425925	0.967516	Virginica	Virginica
0.000110	0.852783	0.776644	Versicolor	Virginica
0.001190	0.345579	0.981445	Virginica	Virginica
0.998808	0.532462	0.039769	Setosa	Setosa
0.999335	0.502827	0.038002	Setosa	Setosa

As can be seen from the table above, the predicted class values match with the ground truth class values which means that the classifiers are working as desired for the test set. Moreover, the predictions are not biased towards a certain class since we can observe all three classes in the table. The interesting thing is that the classifiers are making highly accurate predictions based on the table since each prediction class actually matches their corresponding ground truth class. This is an indication that the model we have chosen, is a well suited model for this dataset, so a linear line can separate different classes in this set.

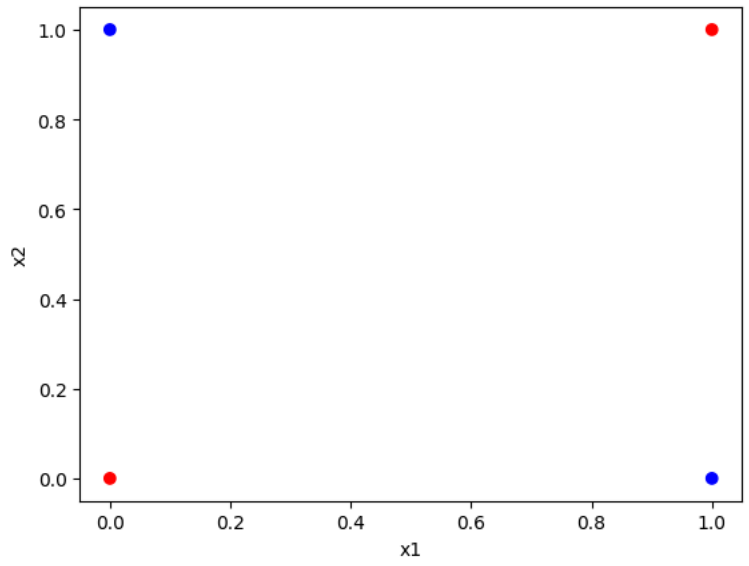
Q8. Calculate the accuracy of the classifier on the test set, by comparing the predicted values against the ground truth. Use a softmax for the classifier outputs. [1 mark]

```
Setosa classifier accuracy: 66.67%
Versicolor classifier accuracy: 70.00%
Virginica classifier accuracy: 63.33%
```

The accuracies of each classifier can be seen from the output above. An accuracy of 66.67% means that the Setosa classifier correctly identified 66.67% of samples as Setosa or not Setosa, based on a threshold applied to the softmax output. A 70.00% accuracy suggests that the Versicolor classifier is moderately effective at distinguishing Versicolor samples from non-Versicolor samples. With an accuracy of 63.33%, this classifier has the lowest success rate among the three.

```
import torch
import torch.nn.functional as F

def calculate_accuracy(model, x, y_true):
    model.eval()
    with torch.no_grad():

        y_pred = F.softmax(model(x), dim=1)
        preds = y_pred.argmax(dim=1)
        accuracy = (preds == y_true).float().mean().item()
    return accuracy

setosa_accuracy = calculate_accuracy(setosa_model, x_test, setosa_testy)
print(f"Setosa classifier accuracy: {setosa_accuracy * 100:.2f}%")

versicolor_accuracy = calculate_accuracy(versicolor_model, x_test, versicolor_testy)
print(f"Versicolor classifier accuracy: {versicolor_accuracy * 100:.2f}%")

virginica_accuracy = calculate_accuracy(virginica_model, x_test, virginica_testy)
print(f"Virginica classifier accuracy: {virginica_accuracy * 100:.2f}%")
```

Q9. Looking at the datapoints below, can we draw a decision boundary using Logistic Regression? Why? What are the specific issues or logistic regression with regards to XOR? [2 marks]



Clearly if we examine the graph above it is not possible to draw a decision boundary using logistic regression, the reason is that Logistic Regression is a linear classifier

that can only generate linear decision boundaries, whereas the XOR problem is not linearly separable as can be observed from the graph. In other words there is no linear decision boundary that can separate the red dots from the blue dots. We will require a more complex decision boundary than a straight line due to the fact that there is a non linear relationship in the XOR problem

Zeynep Kurtulus
Student id: 240037253